Rapid Circle Generation on Two-Dimensional Cellular Automata by $O(r^2)$ Time Algorithm

AUTHOR(S):

Watanabe, Satoru; Okawa, Satoshi

CITATION:

Watanabe, Satoru ...[et al]. Rapid Circle Generation on Two-Dimensional Cellular Automata by $O(r^2)$ Time Algorithm. 数理解析研究所講究録 2014, 1873: 163-169

ISSUE DATE:

2014-01

URL:

http://hdl.handle.net/2433/195500

数理解析研究所講究録
第1873巻 2014年 163-169

Rapid Circle Generation on Two-Dimensional Cellular Automata by $O(r^2)$ Time Algorithm

渡辺 識　　大川 知
Satoru Watanabe　　Satoshi Okawa
会津大学
The University of Aizu

1 Introduction

In a dot matrix method, as a pattern or a figure is obtained by dots on lattices of fixed size, several dot patterns for each pattern should be prepared depending on the screen size. In recent years, several methods to draw some patterns on a screen independent from screen size have been proposed[4][6]. Komatsu has proposed methods to draw a square and a rectangle of maximum size at the center of the screen. Watanabe and Okawa have proposed another method to draw a square and a diamond[6], and a method to draw a circle of radius r at the center of the screen in $O(r^3)$ time[9] .

In this paper, we review the definitions of a pattern and a pattern generation, and we investigate a $O(r^2)$ time algorithm to draw a circle of radius r of maximum size at the center of the screen (two-dimensional cellular automata) by using Firing Squad Synchronization(FSS) method. First, we define two-dimensional patterns as equivalence classes which are obtained by the similarity relation defined by movings and scaling on two-dimensional plane. For an $m \times n$ screen, we define the pattern generation to display with appropriate size (and position) on the screen, and on the discretized screen. Next, we consider a correspondence between the discretized screens and cellular automata, and we define the pattern generation on the cellular automata. Furthermore, we review a basic signal propagation and a square counting method which is used to draw a circle, and FSS on the cellular automata. In the last part, we show a $O(r^2)$ time algorithm to draw a circle of radius r of maximum size at the center of the screen by using the square counting method and FSS method.

2 Pattern Generation

Let $\mathbb{R}$ be a set of real numbers, and a two dimensional plane is denoted by $\mathbb{R} \times \mathbb{R}$. A set $F \subseteq \mathbb{R} \times \mathbb{R}$ is called a two dimensional figure, a set of all two dimensional figures is denoted by $\mathcal{F}$, that is $\mathcal{F} = \{F | F \subseteq \mathbb{R} \times \mathbb{R}\}$. A figure which is obtained with moving F by $d \in \mathbb{R} \times \mathbb{R}$ is denoted by $F + d = \{p + d | p \in F\}$, and a figure which is obtained with extending F by $a (a > 0)$ times is denoted by $a \cdot F = \{a \cdot p | p \in F\}$. We define mappings S_d and Z_a as follows respectively,

$$S_d(F) = F + d \quad \text{and} \quad Z_a(F) = a \cdot F.$$

We define a similarity relation $\sim$ on $\mathcal{F}$ using S_d and Z_a as follows.

For $F_1, F_2 \in \mathcal{F}$, $F_1 \sim F_2 \Leftrightarrow F_2 = S_d Z_a(F_1)$ $(= aF_1 + d)$.

The relation $\sim$ is an equivalence relation on two dimensional figures. We define a pattern as a equivalence class using this relation as follows.

Definition 1
For a figure F, a pattern $[F]$ containing F is defined by

$$[F] = \{F' | F' \sim F\},$$

and a set of pattern $\mathcal{P}$ is defined by

$$\mathcal{P} = \mathcal{F}/\sim = \{P | P = [F], F \in \mathcal{F}\}.$$

For any $m, n > 0$, $[0, m] \times [0, n] \subseteq \mathbb{R} \times \mathbb{R}$ is called a screen of size $m \times n$, and it denoted by $C_{m \times n}$, where $[a, b]$ is an interval $\{x | a \leq x \leq b\}$.

Definition 2
For a pattern $P \in \mathcal{P}$ assuming $P = [F]$, a generation of P on $C_{m \times n}$ is to obtain a set $D \subseteq C_{m \times n}$ such that D satisfies following conditions.

1. $\exists a, d \quad D = S_d Z_a(F)$,
2. $\forall \epsilon, \epsilon' > 0 \quad S_{d+\epsilon} Z_{a+\epsilon'}(F) \not\subseteq C_{m \times n}$.

Following discussion, we assume that m and n are integers for simplicity. When we display a figure on a screen, the screen has to be discretized, so we discretize $C_{m \times n}$ by dividing the width by $m - 1$ and dividing the length by $n - 1$. Then, a copy of a small screen of size 1×1 is set at each lattice point. The small screen at the leftmost and the bottom position of the discretized screen is $c_{0,0}$, and a screen of size 1×1 which is positioned in the ith position from the left side of the array and jth position from the bottom of the array is described by $c_{i,j}$, that is $c_{i,j} = C_{[i-0.5, i+0.5] \times [j-0.5, j+0.5]}$.

We define the screen $C_{m,n}$ which is obtained by discretizing $C_{m \times n}$ as follows,

$$C_{m,n} = \{c_{i,j} | 0 \leq i \leq m, 0 \leq j \leq n, \quad i, j \in N\}.$$

We define the pattern generation on the discretized screen as follows.

Definition 3
For a pattern $P = [F] \in \mathcal{P}$, a generation of P on $C_{m,n}$ is to obtain the following set $D' \subseteq C_{m,n}$,

$$D' = \{c_{i,j} | c_{i,j} \cap D \neq \phi\}.$$

3 Implementation with Cellular Automata

Two-dimensional cellular automata consist of copies of a finite automaton (cell) which are positioned at lattice point. We call $a_{i,j}$ a cell at ith row and the jth column from the leftmost lowest cell. Each cell changes its own state to the state which is determined according to its own state and the adjacent cell's states. We call the own and adjacent cells *neighbors* and the function to determine the next state according to neighbor's states a *local mapping*. A configuration of $\mathcal{M}$ is a global state of $\mathcal{M}$. The global state is determined by the distribution of all cell's states. For time t, when $t = 0$, the configuration is called *initial configuration*. The interval of updating state is called a *step*.

Formally, a two-dimensional cellular automaton $\mathcal{M}$ is defined as follows,

$$\mathcal{M} = (M, Q, \sigma, N),$$

where $M \subset Z \times Z$ is a coordinate set to express positions of cells (we assume it is connected), and Z means a set of integers. Q is a set of states, σ is $Q \times Q^{|N|-1} \rightarrow Q$ is a local mapping, N is a set of neighbors.

In this paper, we investigate the automata which are placed m cells widthways and n cells lengthways, we call them $m \times n$ cellular automata, and we assume N as Neumann neighborhood, namely consisting of the own, upper, lower, right and left cells. In an initial configuration of $\mathcal{M}$, $a_{0,0}$ is active, and all the other cells are in *quiescent*.

By regarding each cell $a_{i,j}$ as $c_{i,j}$ in the discretized screen $C_{m,n}$, the set M can be regarded as the discretized screen $C_{m,n}$, and then, an $m \times n$ cellular automaton can be denoted as follows,

$$\mathcal{M} = (C_{m,n}, Q, \sigma, N).$$

Therefore, we regard a problem to generate P on $C_{m,n}$ as a problem to generate P on a cellular automaton $\mathcal{M}$, that is, the problem is to construct $\mathcal{M}$ which generates P.

Here, to construct such a $\mathcal{M}$ is to provide σ which specifies $D' \subseteq C_{m,n}$ by letting $a_{i,j}$ be in a special state s if $a_{i,j} \in D'$ at a certain time, starting from the initial configuration.

4 Techniques for Circle Generation

We explain some techniques for a circle generation on a two-dimensional cellular array.

4.1 Basic Signal Propagation

Assume a cell is in state s. Then, we call the signal specified by s propagates at speed $1/k$ if the next cell of the cell in s changes its own state to s at k steps. A cell can send signals to all directions upper, lower, right, and left directions.

4.2 Signals to Count Square Steps

To draw a circle pattern, we need to count i^2 for any integer i. We will explain how to count square as follows[8].

Cell a_0 sends Signal s with speed $1/1$ to the right. A cell which receives Signal s for the first time sends Signal $\bar{s}$ with speed $1/1$ to the left, and Signal $\bar{s}$ moves to the left until reaching Cell a_0. Receiving Signal $\bar{s}$, Cell a_0 sends Signal s to the right again. By repeating this procedure, Cell a_i receives Signal s just in i^2 steps as shown in Figure 1.

By following argument, it is clear that this procedure counts i^2 steps for $i > 0$. Assume that a_i receives s for first time at i^2. Then a_i send back $\bar{s}$ to a_0. Receiving Signal $\bar{s}$, a_0 sends Signal s to the right direction again. And then, the next cell a_{i+1} receives Signal s with $2i+1$ steps after that a_i receives Signal s for the first time, that is, a_{i+1} can receive Signal s with $i^2 + 2i + 1 = (i+1)^2$ steps. Therefore, we can count square steps by the procedure.

The procedure which counts square steps initialed from a cell a by using s and $\bar{s}$ is called *Square*(a, s).

4.3 Firing Squad Synchronization

The Firing Squad Synchronization(FSS) Problem for one dimensional cellular array was proposed by J.Myhill. The end cells know that they are located at the end of the array, and the all other cells don't know their own location in the array. At $t = 0$, a cell at the end of the array is in *general* state, and all the other cells are in *quiescent* state. The goal of this problem is to design a set of states and a local mapping that leads all cells to a special state called *firing* simultaneously. At present, an optimal $2n - 2$ steps algorithm is known for the cellular array

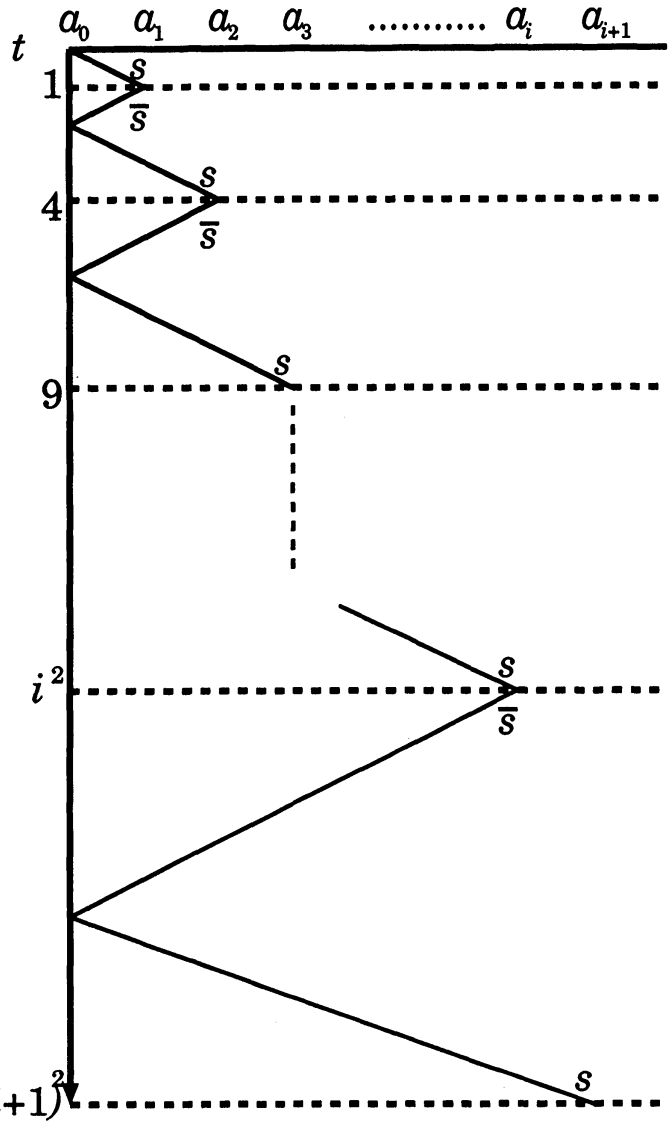

Figure 1: Time-Space diagram of *Square*(a_0, s)

of length n [10]. We use the algorithm for FSS for one dimensional array in order to draw a circle in $O(r^2)$.

5 Circle Generation on Cellular automata

We investigate a method to generate a circle of radius r ($= n/2$) of maximum size at the center of a given $m \times n$ cellular automaton. In the following example, we assume that $m > n$. And then the maximum square $ABCD$ at the center O of the screen, cells at L, R, H, K can be obtained by the method in [6] as shown in Figure 2.

5.1 A Method to Generate a Circle

As describe above, we can assume that cells A, B, C, D, O, L, R, H and K are set as shown in Figure 2 and that as the initial configuration of the circle generation, only cells at O, A, B, C, and D are in active state.

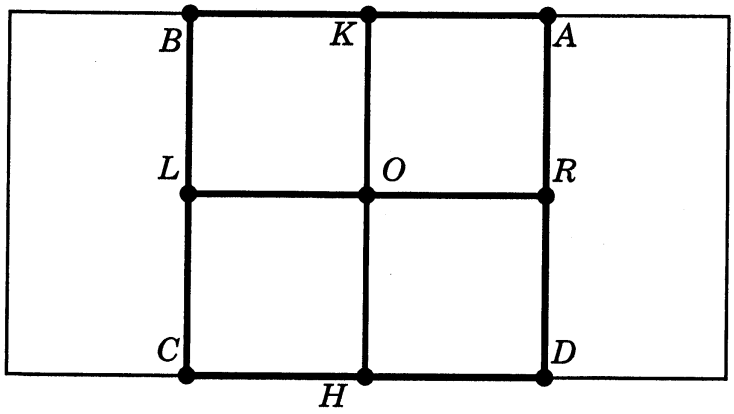

Figure 2: setting the center and corner cells on the screen

We explain process of the generation of a circle at the center of the screen. In the following discussion, we explain how to draw the part of the circle in the first quadrant. We name cells on OR $a_{i,0}$ $(0 < i \leq r)$ (see Figure 3).

We consider cellular automata with two layers. The first layer L_a is used for counting squares and another layer L_b for FSS algorithm.

(1) Implementation of $Square(a_{0,0}, s)$, $Square(a_{i,0}, p)$ on L_a

On the layer L_a, for $a_{i,j}$ $(0 \leq i, j \leq r)$, we will check if $a_{i,j}$ is in a circle by counting $i^2 + j^2$. This counting is performed by $Square(a_{0,0}, s)$ and $Square(a_{i,0}, p)$, that is, first, $Square(a_{0,0}, s)$ counts i^2, and then $Square(a_{i,0}, p)$ starts and continues the counting.

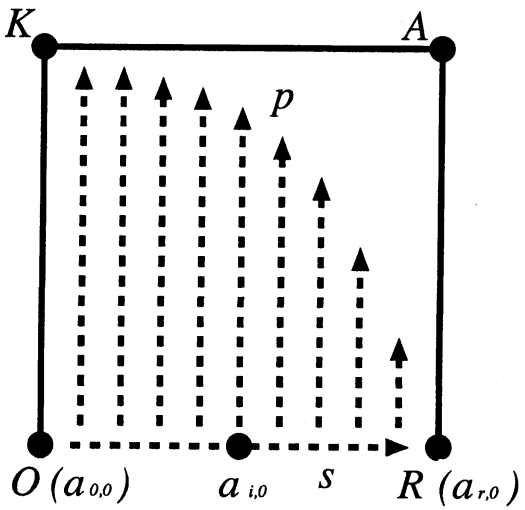

Figure 3: implementation of $Square(a_{0,0}, s)$, $Square(a_{i,0}, p)$ on L_a

(2) Implementation of FSS algorithm on L_b

On the layer L_b, we will implement the FSS algorithm which fires just at r^2 steps, for the cell in the first quadrant. As the number of cells in the first quadrant is $(r+1) \times (r+1)$, we need some tricks. First, we use the FSS algorithm tor one-dimentional version instead of the faster

algorithm for two-dimensional one. To implement it, we make a path as shown in Figure 4. Then, we get two generals at O and A to reduce the steps by factor 2. At last, we adjust two steps by sending signals from O to $a_{1,1}$, and by letting A to wait for two steps to start the algorithm. Then, by applying the FSS algorithm, we can fire on the area of $(r+1) \times (r+1)$ cells just at r^2 steps on L_b.

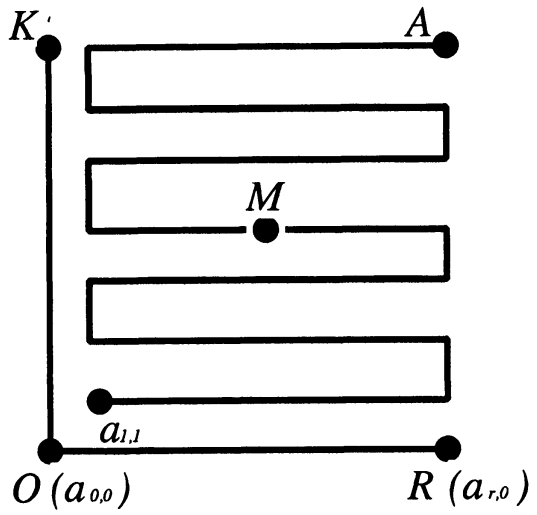

Figure 4: implementation of FSS algorithm on L_b

(3) Circle Generation

Conbining (1) and (2), we can draw a circle. When the FSS finishes on the layer L_b, only cells which have received Signal p on the layer L_a become firing states.

To genrate whole circle, the method described above is performed on all four quadrants. The axes which are between two quadrants turn the firing state after the two quadrants become the firing state on the layer L_b, and then we can obtain the whole circle.

6 Conclusion

In this paper, we review the definition of a pattern and a pattern generation on a screen, and a pattern generation on a discretized screen. Next, we consider a correspondence between the discretized screens and cellular automata, and we defined the pattern generation on the cellular automata. Furthermore, we review FSS problem and its $2n-2$ steps algorithm, and we intoroduce the method *Square*(a, s) to count square to draw a circle. In the last part, we show an $O(r^2)$ time algorithm to draw a circle of a radius r of maximum size at the center of the screen by combining *Square*(a, s) and FSS algorithm. We implemented this algorithm on the cellular automata with two-layer structure. As a result, the algorithm can be described easily. In layer L_a, 8 states are needed to perform *Square*$(a_{0,0}, s)$ and *Square*$(a_{i,0}, p)$. In layer L_b, 20 states are needed to perform the FSS algorithm. By simple multiplication of those numbers, we consider that about 160 states are needed to draw a circle, and we also consider that it is possible to reduce the number of the states by carefull implementation of algorithms on cellular automata.

Acknowledgements

The authors are grateful to the participants of the symposium "Algebras and Computer Science"for their comments to improve the research and its applications.

reference

[1] Y.Mizuno, A New Solution of the Firing Squad Synchronization Problem for Two Dimensional Rectangle Arrays, *University of Aizu, Grad.Thesis.*, March, 2005.

[2] M.Teraoka, et al. A Design of Generalized Optimum-Time Firing Squad Synchronization Algorithm for Two-Dimensional Cellular Arrays, *The 18th Annual Conf.of Japanese Society for Artificial Intelligence*, 3H1-04,2004.

[3] A.Settle and J.Simon, Smaller solutions for the firing squad, *Theoretical Computer Science*, 276, pp.83-109, 2002.

[4] T.Komatsu, Pattern Generation in Two Dimension Plane, *University of Aizu, Grad.Thesis.*, March, 2008.

[5] S.Wolfram, Two-Dimensional Cellular Automata, *Journal of Statistical Physics*, vol 38, p901-946, March 1985.

[6] S.Watanabe and S.Okawa, Pattern Generation on Two Dimensional Cellular Automata, *Forum on Information Technology 2009*, A-023.

[7] J. Mazoyer, A six state minimal time solution to the firing squad synchronization problem, *Theoretical Computer Science*, vol.50, pp.183-238,1987.

[8] Y. Nishitani, Firing Synchronization Problems for Patterns specified by the Sub-General Position, *Cellular Automaton Seminar in University of Aizu* ,2010.

[9] S.Watanabe and S.Okawa, Circle Generation on Two-Dimensional Cellular Automata, *Algebraic Systems and Theoretical Computer Science, RIMS 1809*,pp161-168, 2012

[10] A.Waksman, An optimum solution to the firing squad synchronization problem, *Information and Contorl*, pp66-78, 1966.